REMARKS

The present remarks are in response to the Final Office Action entered in the above identified case and mailed on June 7, 2010. Claims 1-37 are pending in the application. All stand rejected under 35 U.S.C. §103(a). Claims 1, 8-15, 20-26, and 32-27 were rejected over U.S. Patent No. 7,086,009 to Resnick et al. (hereinafter "Resnick") in view of U.S. Patent No. 5,926,177 to Hatanaka et al. (hereinafter "Hatanaka," and claims 2-7, 16-19 and 27-31 were rejected over Resnick in view of Hatanaka and further in view of U.S. Patent Number 5,485,600 to Joseph et al. (hereinafter "Joseph"). The grounds for rejection presented in the Final Office Action are substantially the same as the grounds for rejection presented in the previous Non-Final Office Action. Because Applicants believe that both the previous Non-Final Rejection and the present Final Rejection are based on a faulty reading of Resnick, Applicants respectfully traverse and request that the Examiner reconsider the final rejection and allow the claims to issue.

It is well established that a claim is unpatentable under 35 U.S.C. §103(a) only if each and every element of the claim is taught or suggested by the prior art. As argued in Applicants previous response and as reiterated here, the combined teachings of Resnick and Hatanaka do not teach or suggest every element of any of claims 1, 8-15, 20-26, and 32-27, and the combined teachings of Resnick, Hatanaka and Joseph do not teach or suggest every element of claims 2-7, 16-19 and 27-31.

Claims 1, 8-15, 20-26 and 32-37

Independent claims 1, 15, and 26 each stand rejected under 35 U.S.C. §103(a) over the combined teaching of Resnick & Hatanaka. Independent claims 1, 15, and 26 include many similar features and are allowable over Resnick and Hatanaka for similar reasons. Accordingly, Applicants will argue the patentability of independent claim 1 at length

followed by a brief discussion of any additional or different arguments in favor of the patentability of claims 15 and 16.

Claim 1 calls for a display entity for use in presenting a visual depiction of a process entity of a process plant to a user on a display device. The display entity comprises a computer readable memory and a display object stored on the computer memory. The display object is adapted to be executed on a processor. The display object includes a property memory adapted to store a value of a property associated with the process entity and a graphic representation of the process entity adapted to be displayed to a user on a display device when the display object is executed on the processor. Claim 1 further calls for a definition routine adapted to enable a user to define a routine that operates in conjunction with the graphic representation of the process entity and the value of the property associated with the process entity during execution of the display object. Finally, claim 1 calls for a routine that operates in conjunction with the graphic representation of the process entity to alter the manner in which the graphic representation of the process entity is displayed to the user according to the value of the property to reflect an operating condition of the process plant associated with the process entity.

The Examiner's assertions to the contrary not withstanding, Resnick does not disclose a number of the features called for in independent claim 1 of the present application. First, and most generally, Resnick does not teach or suggest a display object at all. More specifically, Resnick does not teach or suggest a display object that includes a property memory adapted to store a value of a property associated with a process entity, a graphic representation of the process entity adapted to be displayed to a user on a display device, a definition routine adapted to enable a user to define a routine that operates in conjunction with the graphic representation of the process entity and the value of the property associated

with the process entity during execution of the display object, and a routine that operates in conjunction with the graphic representation of the process entity to alter the manner in which the graphic representation of the process entity is displayed to the user according to the value of the property to reflect an operating condition of the process plant associated with the process entity.

According to the Examiner, Resnick Fig. 11 shows a display entity for use in presenting a visual depiction of a process entity of a process plant to a user on a display device. While Fig. 11 does appear to show visual representations of process control entities displayed on a display device, a careful reading of Resnick shows that Fig. 11 does not represent a display device at all. In the Brief Description of the Drawings Fig. 11 is described as a schematic diagram depicting the simple plant process components logically grouped into areas. (Col. 4, lines 66-67). Resnick Fig. 11 is further described as a logical grouping of related process components into areas. (Col. 22, lines 5-6). The illustration depicts the system as a series of areas comprising logically grouped controlled process components. (Col. 22, lines 7-9). The areas shown in Resnick Fig. 11 include a raw material store area 1100, a production area 1102, a line area 1104, and a finished products area 1110. (Col. 22, lines 9-18). Each of the areas includes various process components such as hoppers, mixers, conveyors, etc.

Thus, while the schematic diagram of Resnick Fig. 11 clearly depicts a number of process components, the process components shown in Fig. 11 do not comprises display objects for use in presenting a visual depiction of a process entity to a user on a display device. Rather, Fig. 11 is merely an illustration depicting how various process plant components can be grouped into logical areas according to Resnick's supervisory process control system.

It is not at all clear what component or feature of Fig. 11 the Examiner considers to be a display entity for use in presenting a visual depiction of a process entity of a process plant to a user on a display device. The text describing Fig. 11 in the detailed description of the Invention does not describe a "display entity" for presenting visual depictions of a process entities on a display device. In fact, Resnick never describes any means of actually presenting the process components shown in Fig. 11 to a user during execution of Resnick's supervisory process control system.

Next, the Examiner points to Resnick, col. 7, lines 30-48 as teaching a display object stored on a computer readable memory and adapted to be executed by a processor. The only "object" mentioned in the cited passage, however, is an "application object." Nothing in the cited passage indicates that the application objects comprise display objects for displaying a graphic representation of a process entity during runtime. In fact, Resnick col. 7, lines 30-48 never mentions the word display at all.

Resnick col. 7, lines 30-48 describes deploying a process control and manufacturing application. According to the cited passage, the process control and manufacturing application may be deployed across many different physical computing systems. Accordingly, Resnick describes "a second type of system view," referred to as a deployment model. The deployment model allows a user to configure physical PCs and devices with regard to the application. By configuring the physical machines that will implement the application, the user defines the areas that will run on particular engines thereby determining where the particular application software will be physically executed. Application engines host application objects via a logical grouping object referred to as an "area." The engines are in turn hosted by platform objects at the next lower level of the supervisory process control and manufacturing information application. (Resnick col. 9, lines 14-18.) According to

Resnick col. 7, lines 30-48 a supervisory process control and manufacturing information system provides a configuration view of a deployment model showing the hierarchy with physical PCs, and the areas and application objects running on the physical PCs.

Resnick never describes the application objects or the engine objects, or any other object, as "display objects." Furthermore, various examples of hierarchical tree structures or models are shown in Resnick Figs. 12, 13, 14B, and 15-17. None of these figures show display objects having the features of a display object as called for in claim 1 of the present application. In fact, Resnick col. 7, lines 30-48 is completely silent regarding display objects and is completely irrelevant to the subject matter of claim 1.

Next, the Examiner points to Resnick col. 11, lines 29-67 as teaching a display object that includes a property memory adapted to store a value of a property associated with the process entity that the claimed display entity is used to visually represent. Yet again, the cited passage is completely devoid of any such teaching.

Resnick col. 11, lines 29-67 is a lengthy passage spanning 3 paragraphs. The first two paragraphs describe engine objects and the third describes application objects. The cited passage is found within a portion of the disclosure describing Fig. 2. Resnick Fig. 2 is a class diagram that depicts a hierarchical arrangement of layered software associated with a computer executing at least a portion of a supervisory process control and manufacturing information application. *Resnick*, col. 10, lines 63-67. According to col. 11, lines 29-67, the engine objects host a set of application objects (the next lower layer of the hierarchy shown in Fig. 2). The application objects implement supervisory process control and/or manufacturing information acquisition functions associated with an application. The engine objects initiate startup and schedule execution of the application objects with the help of a scheduler object. The engine objects maintain a name binding service that binds attribute references (e.g.

tank1.value.pv) to a proper one of the application objects. Ultimately, the engine objects control how execution of application objects will occur.

The application objects described in the third paragraph of the passage cited by the Examiner include a wide variety of objects that execute business logic facilitating the carrying out of a particular process control operation (e.g. turning on pump, opening a valve, etc.) and/or an information gathering/management function. Examples of application objects include: analog input, discrete device, and PID application objects.

Nothing in Resnick col. 11, lines 29-67 describes a display object including a property memory adapted to store a value of a property associated with a process entity. As already argued, Resnick does not disclose display objects at all. Further Resnick col. 11, lines 29-67 does not disclose a display object including a property memory adapted to store a value of a property associated with the process entity. The closest thing resembling a property memory adapted to store a value of a property associated with a process entity described in the cited passage is the sentence beginning at col. 11, line 44, which states "The engine objects maintain a name binding service that bind attribute references (e.g., tank1.value.pv) to a proper one of the application objects." This vague allusion to process entity property values, however, says nothing about a property memory included as part of a display object. At most it indicates that attribute references are bound to application objects. Since application objects are not equivalent to display objects, and because binding attribute references to application objects does not necessarily suggest a property memory, it cannot be said that this sentence teaches or suggests a property memory adapted to store a value of a property associated with a process entity as called for in claim 1 of the present application.

If the Examiner considers some other portion of Resnick col. 11, lines 29-67 as teaching a property memory adapted to store a value of a property associated with the process

entity, the Examiner is requested to point out explicitly where this feature can be found and to explain in detail how it relates to a property memory included in a display object as claimed in claim 1 of the present application.

Next, the Examiner points to Resnick col. 12, lines 5-11 as teaching a graphic representation of a process entity adapted to be displayed to a user on a display device when the display object is executed on a processor. Yet again, the cited passage is completely irrelevant to the claimed subject matter it is cited against. Resnick col. 12, lines 5-11 describes an exemplary embodiment that includes an application interface that is accessed by engine objects and schedulers. The engine objects access the application object interface to initialize an application object, startup an application object, and shutdown an application object. The schedulers use the application object interface to initiate a scheduled execution of the application object.

Resnick col. 12, lines 5-11 does not describe a display device. Rather Resnick col. 12, lines 5-11 describes an interface. This interface is not a user interface. It is an interface between engine objects and schedulers. The engine objects and schedulers are not display objects. Furthermore, the engine objects, schedulers and application interfaces do not include graphic representations of process entities which are displayed to a user on a display device when a display object is executed on a processor. In short, nothing described in Resnick col. 12, lines 5-11 remotely resembles a graphic representation of a process entity that is to be displayed to a user on a display device when a display object is executed on a processor.

Next, the Examiner cites Resnick col. 29, line 61 – col. 30, line 26 as teaching a definition routine adapted to enable a user to define a routine that operates in conjunction with the graphic representation of the process entity and the value of the property associated with the process entity during execution of the display object; Resnick col. 30, line 50 – col.

31, line 16 as teaching a routine that operates in conjunction with the graphic representation of the process entity displayed to the user and that is associated with the value of the property; and Resnick col. 2, lines 21-26 as teaching a routine that operates in conjunction with the graphic representation of the process entity to reflect an operating condition of the process plant associated with the process entity. Yet again the actual teaching of the cited passages falls far short of the scope attributed them by the Examiner.

Applicants distinguished claim 1 over the teaching of Resnick col. 29, line 61 – col. 30, line 26 and Resnick col. 30, line 51 – col. 31, line 16 in the REMARKS section of the AMENDMENT IN RESPONSE TO NON-FINAL OFFICE ACTION filed March 9, 2010. Applicants reassert the arguments from the previous response here. For the sake of brevity Applicants will limit the present discussion to addressing the Examiner's Response to Arguments included in the Final Office Action. For a full discussion as to why Resnick col. 29, line 61 – col. 30, line 26 and col. 30, line 51 – col. 31, line 16 do not teach or suggest the features of claim 1 attributed to them by the Examiner, the Examiner is referred back to the AMENDMENT IN RESPONSE TO NON-FINAL OFFICE ACTION filed on March 9.

In response to Applicant's previous arguments the Examiner correctly notes that Resnick col. 29, line 61 – col. 30, line 26 discloses interface component definitions 2002 including a configuration interface 2010, a runtime interface 2012, and a package interface 2014 to enable interaction with different environments. According to the Examiner the configuration interface 2010 and package interface 2014 each enable an object configuration, editing, packaging, importing, exporting, uploading, downloading, copying and storage. Unfortunately, the Examiner's description of the configuration interface 2010 and the package interface 2014 leaves out an important detail. Resnick col. 30, lines 1-6 actually says: "the Configuration Interface 2010 supports a set of methods enabling an object's

configuration to be edited by the IDE (Integrated Development Environment). The package Interface 2014 supports a set of methods enabling the object to be packaged and then imported, exported, uploaded, downloaded, copied, and/or stored within the configuration database." As one would expect, the configuration interface 2010 is related to configuring an application object. The package interface 2014 relates to preparing the application objects to be manipulated (i.e. imported, exported, uploaded, downloaded, copied and/or stored). The Configuration Interface 2010 and the Package Interface 2014 have absolutely nothing to do with defining a routine that operates in conjunction with a graphic representation of a process entity and the value of a property associated with the process entity during execution of a display object.

Next, the Examiner notes that the runtime interface 2012 further sets methods that provide behaviors to the object during runtime. The precise quote from the specification is as follows. "The Runtime Interface 2012 supports a set of methods executed on identified attributes that provide behaviors to the object during runtime when an object is deployed on a host application engine." Nothing in the cited passage, or elsewhere in Resnick's disclosure, however, indicates that the "object" referred to in the above quote relates to a display object that includes, among other things, a graphic representation of a process entity that is adapted to be displayed on a display device when the display object is executed on a processor. There is nothing in Resnick's disclosure that indicates that the methods executed on identified attributes that provide behaviors to an object during runtime operate in conjunction with a graphic representation of a process entity to alter the manner in which the graphic representation is displayed to reflect an operating condition of the process plant as called for by claim 1 of the present application.

Finally, the Examiner cites Resnick col. 30, line 50 – col. 31, line 16 as teaching a routine that operates in conjunction with a graphic representation of a process entity displayed to a user and that is associated with the value of a property. According to the Examiner an Object Designer disclosed by Resnick supports an object template development process. The Examiner interprets this as a routine that operates in conjunction with a graphic representation of a process entity. The Examiner's "interpretation," however, is inapposite. The Examiner, may interpret white as black, but that does not make it so. Resnick col. 30, line 50 – col. 31, line 16 simply does not teach or suggest a routine that operates in conjunction with a graphic representation of process entity to alter the manner in which the graphic representation of the process entity is displayed to the user according to the value of the property to reflect an operating condition of the process plant associated with the process entity as called for in claim 1.

Resnick col. 30, line 50 – col. 31, line 16 describes a graphical utility that enables a developer to develop objects and primitives for objects. Generally speaking, the Object Designer allows the developer to configure the attributes and primitives associated with an object. The Object Designer allows the developer to specify attribute values and to define entirely new attributes. A basic mode facilitates the creation of application objects from common primitives, custom primitives and utility primitives. Whereas an advanced mode facilitates the creation of new primitives, or a hierarchy of primitives, and/or an object comprising a hierarchy of primitives. Finally, a set of code wizards generate the source code in a specified language based on an object or primitive definition rendered by the Object Designer. In the case of C++ code an attribute class is provided and a custom object class containing attributes are generated. In the case of Visual Basic code, an attribute class is

provided and a customer object class containing attributes is generated. The attribute class contains Get and Set Property routines that access the proper row of an attribute table.

Nothing in the cited passage describes anything remotely related to a graphical representation of a process entity. The Examiner "interprets" the Object Designer as disclosing a routine that operates in conjunction with a graphic representation of a process entity. But this statement begs the question, what in Resnick's disclosure comprises a process entity? And where does Resnick disclose or even suggest a display object that includes a graphic representation of a process entity? Absent a teaching of a display object that includes a graphic representation of a process entity, the Examiner's "interpretation" that the object designer discloses a routine that operates in conjunction with a graphical representation of a process entity is meaningless. According to the Examiner's logic the Object Designer operates in conjunction with something that is never described or suggested by Resnick.

A careful reading of Resnick makes it clear that Resnick does not teach or suggest many of the elements of claim 1 of the present application ascribed to it by the Examiner. It is impossible to reach the conclusion that Resnick teaches or suggests a display object stored on a computer readable memory that is adapted to be executed on a processor, wherein the display object includes a property memory adapted to store a value of a property associated with a process entity, a graphic representation of the process entity adapted to be displayed to a user on a display device when the display object is executed on a processor, a definition routine adapted to enable a user to define a routine that operates in conjunction with the graphic representation of the process entity and the value of the property associated with the process entity during execution of the display object, and a routine that operates in conjunction with the graphic representation of the process entity to alter the manner in which

the graphic representation of the process entity is displayed to the user according to the value of the property to reflect an operating condition of the process plant associated with the process entity, as called for in claim 1 of the present application.

Hatanaka col. 3, line 58 – col. 4, line 45 is cited against claim 1 as disclosing altering the manner in which a graphic representation a process entity is displayed to a user according to the value of a property. It is true that Hatanaka col. 3, line 58 – col. 4, line 45 teaches altering a display model when the state of an object changes. The cited passage does not, however, teach or suggest a display object including a property memory adapted to store a value of a property associated with a process entity, a graphic representation of the process entity, a definition routine adapted to enable a user to define a routine that operates in conjunction with the graphic representation of the process entity, and the value of the property associated with the process entity during execution of the display object as called for in claim 1 of the present application, and which are not taught or suggested by Resnick. Thus, even if Hatanaka does teach the features suggested by the Examiner, the combined teaching of Resnick and Hatanaka nonetheless still fails to teach or suggest every element of claim 1. Accordingly, claim 1 and the claims depending therefrom are not unpatentable over Resnick and Hatanaka under 35 U.S.C. §103(a) and should be allowed.

As mentioned above, independent claim 15 was rejected over the combined teaching of Resnick and Hatanaka on similar grounds as claim 1. Claim 15, however, is actually allowable over the cited references for reasons similar to claim 1. Resnick and Hatanaka do not teach or suggest a library of graphic objects, each graphic object including a visual representation of a physical or a local entity within the process plant, a graphically based editor canvas routine that enables a user to define an executable graphic display by placing one or more visual representations of the graphic objects from the library of graphic objects

onto an edit-canvas to define a manner in which the one or more visual representations of the graphic objects will be displayed on a display device to a user during execution of the graphic display, a property definition canvas routine adapted to enable a user to define a property associated with at least one of the plurality of graphic objects, and a binding definition routine adapted to enable a user to specify a binding between the property and a runtime environment within the process plant.

The Examiner points to Resnick Fig. 10 as teaching a library of graphic objects including a visual representation of a physical or logical entity within a process plant. Like Fig. 11, however, Fig. 10 is merely a schematic diagram showing various process components in an exemplary process plant. Nothing in Resnick's disclosure indicates that Fig. 10 represents a visual display, or that the individual process components depicted in Fig. 10 represent a library of graphic objects.

The Examiner further points to a base object editor 2034 shown in Resnick Fig. 20 as teaching a graphically based editor canvas routine that enables a user to define an executable graphic display by placing one or more visual representations of the graphic objects from the library of graphic objects onto an edit-canvas to define a manner in which the one or more visual representations of the graphic objects will be displayed on a display device to a user during execution of the graphic display. The base object editor 2034 is described at Resnick col. 32, lines 12-14. The description of the base object editor is contained in a single sentence: "A base object editor 2034 provides standard editor pages common to all objects." That is the only mention of a base object editor in all of Resnick's disclosure. The Examiner's ability to envision a graphically based editor canvas routine that enables a user to define an executable graphic display by placing one or more visual representations of the graphic objects from the library of graphic objects onto an edit-canvas to define a manner in

which the one or more visual representations of the graphic objects will be displayed on a display device to a user during execution of the graphic display from a single sentence of just 12 words is an amazing feat of imagination. More likely, the Examiner is using hindsight to read detailed elements of Applicant's claim into a vague reference to a base object editor. The mere mention of standard editor pages common to various objects, does not teach or suggest the graphically based editor canvas routine having the characteristics called for in claim 15 of the present application.

Finally, the Examiner points to interface definitions 2002 (also shown in Fig. 20) as teaching a property definition canvas routine adapted to enable a user to define a property associated with at least one of the plurality of graphic objects. The interface definitions are described at Resnick col. 29, line 62 – col. 30, line 6. The interface definitions include a Configuration (editor) interface, a Runtime Interface and a Package Interface. Nothing in the description of the Configuration interface, Runtime interface or the Package interface, describes a property definition canvas routine. Resnick does describe GetAttribute and SetAttribute methods that are supported by the Runtime interface that are executed on identified attributes that provide behaviors to objects during runtime. The mere mention of various methods that are supported by an interface that may be applied to objects to call and set various attributes of an object to affect its behavior during runtime is not equivalent to the disclosure of a property definition canvas routine that is adapted to enable a user to define a property of the graphic objects (as opposed to merely setting an attribute). Because neither Resnick nor Hatanaka teach or suggest these features of independent claim 15, claim 15 and the claims depending therefrom are not unpatentable under 35 U.S.C. §103(a) and should be allowed.

Next Applicants turn to independent claim 26. Claim 26 was rejected on the same grounds as claim 1. According to the Examiner Resnick discloses a processor; a plurality of graphical objects adapted to be executed by the processor, and a graphical display. However, as already argued with respect to claim 1, Resnick does not in fact disclose a graphical display that includes a plurality of graphical objects, wherein the graphical objects include a property memory adapted to store a value for a property associated with at least one of the plurality of graphical objects, a definition routine adapted to enable a user to define a routine that operates in conjunction with execution of at least one of the graphic objects during execution of the visual display and a routine that operates in conjunction with execution of at least one of the graphical objects when the graphical display object is executed by the processor to alter the manner in which the at least one of the graphical objects is presented within the graphical display object according to the value of the property to reflect an operating condition of the process plant associated with the process entity. Because neither Resnick nor Hatanaka teach or disclose these features of independent claim 26, claim 26 and the claims depending therefrom are not unpatentable under 35 U.S.C. §103(a) over Resnick and Hatanaka and should be allowed.

Applicants now turn to the rejection of claims 2-7, 16-19, and 27-31 as being unpatentable over Resnick and Hatanaka and further in view of Joseph. Applicants note the claims 2-7 all depend either directly or indirectly from independent claim 1. Claims 16-19 all depend either directly of indirectly from independent claim 15. Finally, claims 27-31 all depend either directly or indirectly from independent claim 26. Joseph is cited merely for teaching an animation routine that animates a graphic representation in response to changes in data values. Even if Joseph does teach such a feature, however, the combined teaching of Resnick, Hatanaka and Joseph still fails to teach or suggest the features of the independent

claims absent from the disclosures of Resnick and Hatanaka as described above. Therefore even if Joseph teaches a routine that animates graphic representations, the art of record nonetheless fails to teach or suggest every element of the invention claimed in claims 2-7, 16-19 and 27-31. Accordingly, claims 2-7, 16-19, and 27-31 are not unpatentable under 35 U.S.C. §103(a) over Resnick, Hatanaka and Joseph and should be allowed.

CONCLUSION

In view of the arguments presented above Applicants respectfully submit that all of the claims pending in the application are in condition for allowance. Applicant's therefore request that the Examiner withdraw the final rejection and allow the application to issue. If the Examiner has any questions regarding the present response, however, the Examiner is encouraged to call Applicants' attorney at the number provided below.

Dated: September 16, 2010

Respectfully submitted,

By /Jeffrey H. Canfield #38,404/
Jeffrey H. Canfield
Registration No.: 38,404
MARSHALL, GERSTEIN & BORUN LLP
233 S. Wacker Drive
6300 Willis Tower
Chicago, Illinois 60606-6357
(312) 474-6300
Attorney for Applicant